

# Constrained Distributed Algebraic Connectivity Maximization in Robotic Networks <sup>★</sup>

— TECHNICAL REPORT —

Andrea Simonetto <sup>a</sup>, Tamás Keviczky <sup>a</sup>, and Robert Babuška <sup>a</sup>

<sup>a</sup>*Delft Center for Systems and Control, Delft University of Technology, Mekelweg 2, 2628 CD Delft, The Netherlands*

---

## Abstract

We consider the problem of maximizing the algebraic connectivity of the communication graph in a network of mobile robots by moving them into appropriate positions. We define the Laplacian of the graph as dependent on the pairwise distance between the robots and we approximate the problem as a sequence of Semi-Definite Programs (SDP). We propose a distributed solution consisting of local SDP's which use information only from nearby neighboring robots. We show that the resulting distributed optimization framework leads to feasible subproblems and through its repeated execution, the algebraic connectivity increases monotonically. Moreover, we describe how to adjust the communication load of the robots based on locally computable measures. Numerical simulations show the performance of the algorithm with respect to the centralized solution.

*Key words:* Distributed Control of Robotic Networks, Connectivity Maximization, State-dependent Graph Laplacian, Collaborative systems, Networked robotics

---

## 1 Introduction

Teams of autonomous mobile robots that communicate with one another to achieve a common goal are considered in several applications ranging from underwater and space exploration [12, 17], to search and rescue [6, 16], monitoring and surveillance [5, 18]. These robots possess on-board processing capability, but the common task can only be achieved through information exchange among the members and possibly a base station. Such multi-robot teams are thus often referred to as robotic networks. Among the engineering and research questions these applications pose, maintaining connectivity between the individual robots and increasing the communication quality under given constraints, have fundamental importance. Several types of coordination and control frameworks that have been recently proposed rely on agreement protocols or consensus processes that lead to coordinated team actions [4, 13, 21]. Since these protocols typically assume only local communication among “neighboring” robots, the interconnection topology of the underlying communication graph influences greatly

their effectiveness. In particular, their convergence properties are dictated by the algebraic connectivity of the communication graph [19].

In this paper, we study distributed solutions for maximizing the algebraic connectivity of the communication graph (often denoted as  $\lambda_2$ ) in mobile robotic networks. We note that, besides the benefit in terms of improved communication, the tools that we develop are instrumental for handling cases where network of mobile robots have other common tasks, in addition to the requirement to increase their  $\lambda_2$ . Examples of scenarios where our solution could, or has been used in a preliminary version, are collaborative multi-target tracking [11, 23] and coordination control [10]. For example, in [11], the authors specifically increase the  $\lambda_2$  of a special weighted graph that describes the visual connection with multiple targets. Their aim is to move a group of mobile robots in order to increase the visibility of multiple targets. In this context, the problem of  $\lambda_2$  maximization could also be seen as an alternative formulation of the optimal sensing placement problem in a dynamic environment [11].

References [7, 22, 25–27, 29] give a comprehensive overview of distributed algorithms for robotic networks that aim at ensuring connectivity (i.e., nonzero  $\lambda_2$  rather than its maximization). Typically, these algorithms are either limited to specific scenarios only, or imply heavy communication requirements, and often they are not di-

---

<sup>★</sup> This paper was not presented at any IFAC meeting. Corresponding author A. Simonetto.

*Email addresses:* [a.simonetto@tudelft.nl](mailto:a.simonetto@tudelft.nl) (Andrea Simonetto), [t.keviczky@tudelft.nl](mailto:t.keviczky@tudelft.nl) (Tamás Keviczky), [r.babuska@tudelft.nl](mailto:r.babuska@tudelft.nl) (Robert Babuška).

rectly related to the solution of the centralized version. In terms of distributed connectivity maximization, the available literature appears to be very limited. To the best of our knowledge, only the work in [9] investigates a distributed solution for the maximization of  $\lambda_2$  based on a simplified scenario where the dynamics of the robots are represented by a single integrator and no constraints are present. The authors use a two-step distributed algorithm, which relies on super-gradients and potential functions. The required communication load scales with the square of the graph diameter which may impede fast real-time implementations for large groups of robots.

We consider as starting point the centralized optimization procedure of [3, 11, 14]. In these works the maximization of the algebraic connectivity is approximated as a sequence of Semi-Definite Programs based on the notion of state-dependent graph Laplacian, while the agents are modeled as discrete-time single integrators.

Our first contribution is to modify the aforementioned centralized optimization procedure in order to handle more generic LTI robot dynamics. The resulting optimization problem is then proven to be feasible at each time step under quite general assumptions.

As our second contribution, we propose a distributed solution for the centralized problem (Algorithm 1) substantially extending our preliminary results in [24]. Our proposed distributed approach relies on local problems that are solved by each robot using information only from nearby neighbors and, in contrast with [9], it does not require any iterative schemes, making it more suitable for real-time applications. This last property is not a trivial aspect when using common decomposition methods for optimization [1], as done in various approaches to distributed control [15, 20]. In our approach *(i)* we formulate local problems of small size that are clearly related to the centralized one, *(ii)* the *linearized* algebraic connectivity of the approximate problem is guaranteed to be monotonically increasing, *(iii)* the overall optimization scheme is proven to be feasible at each time step under quite general assumptions, and in particular *(iv)* the local solutions are feasible with respect to the constraints of the original centralized problem.

Finally, we characterize the local relative sub-optimality of the optimized  $\lambda_2$  with respect to a larger neighborhood size and we use this characterization to enable each robot to increase or decrease its communication load on-line, while respecting the properties *(ii)* - *(iv)*. This means that our solution can be adapted based on available resources, augmenting or reducing the required communication and computational effort.

The proposed distributed solutions can be seen as a complementary approach to standard subgradient algorithms [1]. Distributed versions of incremental subgradient algorithms are typically communication intensive it-

erative algorithms, in which at each iteration, each agent has to evaluate only a local subgradient of a certain function. Our proposed solutions lie on the other side of the “communication-computation” trade-off spectrum. In fact, each robot solves a reasonably complex convex optimization problem, while the communication among them remains limited. In this context, multi-robot systems embedded with reasonable processing capabilities, where real-time applicability is a strong requirement, could benefit more from our proposed approach than from standard subgradient algorithms.

The paper is organized as follows. Sections 2-3 formulate the approximate centralized problem based on [14]. Starting from a general time-invariant non-convex formulation (6), first we discuss the sequential Semi-Definite Programming approach (13) considering single integrator dynamics for the agents (1), as done in the literature [11, 14]. Second, in Section 4 we extend this sequential Semi-Definite Programming approach to more general LTI agent dynamics (14) in problem (22). The proposed distributed approach for problem (22) is described in Section 5 in problem (25) and Algorithm 1. Its properties are analyzed in Section 6, while the local relative sub-optimality measures are the topic of Section 7. Numerical simulations are shown in Section 8 to assess the performance of the distributed solutions. Conclusions and open issues are discussed in Section 9.

## 2 Problem Formulation

The notation is standard: for any real scalar  $s$ ,  $s \in \mathbb{R}_0$  if  $s \neq 0$ ,  $s \in \mathbb{R}^+$  if  $s \geq 0$ , and  $s \in \mathbb{R}_0^+$  if  $s > 0$ . The matrices  $I_n$  and  $0_n$  represent the identity and the null matrix with dimension  $n \times n$ , respectively. The column vectors  $\mathbf{1}_n$  and  $\mathbf{0}_n$  define vectors of dimension  $n$  where all the entries are 1 and 0, respectively.

Consider a network of  $N$  agents with communication and computation capabilities and express as  $a_i(k)$  the value of the variable  $a$  for agent  $i$  at the discrete time instant  $k$ . The position of agent  $i$  is denoted by  $x_i(k) \in \mathbb{R}^3$  and its velocity by  $v_i(k) \in \mathbb{R}^3$ . In order to introduce the works of [3, 11, 14], we assume the agents to move according to the following discrete-time dynamical system:

$$x_i(k+1) = x_i(k) + v_i(k)T_s \quad (1)$$

where  $T_s$  is the sampling time. This single-integrator model will be extended in subsequent sections.

Graph-theoretic notions are used to model the network. Let  $x(k)$  be the stacked vector containing the positions of the agents, i.e.  $x(k) = (x_1^\top(k), \dots, x_N^\top(k))^\top$ . The set  $\mathcal{V}$  contains the indices of the mobile agents (nodes), with cardinality  $N = |\mathcal{V}|$ . The set  $\mathcal{E}$  indicates the set of communication links. The graph  $\mathcal{G}$  is then expressed as  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  and it is assumed undirected. Let the agent

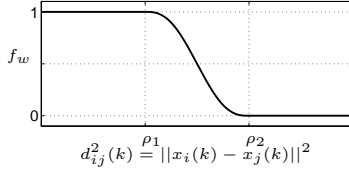


Fig. 1. Weighting function  $f_w(\cdot)$  for modeling connectivity between two agents  $i, j$ . If  $d_{ij}^2(k) < \rho_1$  then  $w_{ij} = 1$ , while if  $d_{ij}^2(k) > \rho_2$  then  $w_{ij} = 0$ .

clocks be synchronized, and assume perfect communication (no delays or packet losses). The agents with which agent  $i$  communicates are called neighbors and are contained in the set  $\mathcal{N}_i$ . Note that agent  $i$  is not included in the set  $\mathcal{N}_i$ . We define  $\mathcal{N}_i^+ = \mathcal{N}_i \cup \{i\}$  and  $N_i = |\mathcal{N}_i^+|$ . Define the Laplacian matrix  $L$  associated with  $\mathcal{G}$  via its entries  $\ell_{ij}$  as  $\ell_{ij}(k) = 0$  for  $(i, j) \notin \mathcal{E}$ ,  $\ell_{ij}(k) = -w_{ij}(k)$  for  $(i, j) \in \mathcal{E}$ , and  $\ell_{ij}(k) = \sum_{l \neq i} w_{il}(k)$  for  $i = j$ . The weights  $0 \leq w_{ij} \leq 1$  are assumed to depend on the squared Euclidean distance of  $x_i(k)$  and  $x_j(k)$  defined as

$$d_{ij}^2(k) = f_d(x_i(k), x_j(k)) = \|x_i(k) - x_j(k)\|^2 \quad (2)$$

and

$$w_{ij}(k) = f_w(\|x_i(k) - x_j(k)\|^2) \quad (3)$$

where  $f_w : \mathbb{R}^+ \rightarrow [0, 1]$  is a smooth nonlinear function with compact support. The weights model the connection strength between two agents. The closer two agents are, the closer to one is the weight, representing an increase in the communication “quality”. For simulation purposes we use the function qualitatively represented in Figure 1, which is one when the squared distance is less than  $\rho_1$  and it is zero when the squared distance is greater than  $\rho_2$ . For a detailed discussion on the choice of  $f_w$  the reader is referred to [14]. As a direct consequence of the above definitions, the entries of the Laplacian matrix  $L$  depend on the state of the agents, making it state-dependent, which we will denote by  $L(x(k))$ .

We are interested in maximizing the algebraic connectivity of the weighted graph by controlling the state of the agents, i.e., moving them to appropriate positions. First of all, we notice that [3]

$$\max_x \lambda_2(x) \equiv \{\max_{x, \gamma} \gamma \text{ s.t. } L(x) + \mathbf{1}_N \mathbf{1}_N^T \succ \gamma I_N\},$$

which can be proven formally as follows.

**Proposition 1** *For any two scalars  $\lambda > \bar{\lambda}_2 > 0$ , the constraint*

$$\lambda_2(L) > \bar{\lambda}_2, \quad (4)$$

*can be formulated with the equivalent Matrix Inequality*

$$L + (\lambda/N) \mathbf{1}_N \mathbf{1}_N^T \succ \bar{\lambda}_2 I_N. \quad (5)$$

*Proof.* By construction, the Laplacian matrix  $L$  has as eigenvector  $\mathbf{e}_1 = \mathbf{1}_N$ . All the other eigenvectors,  $\mathbf{e}_i$ , are orthogonal to  $\mathbf{1}_N$ , meaning  $\mathbf{1}_N^T \mathbf{e}_i = 0$ , for  $i = 2, \dots, N$ . This implies that

$$(L + (\lambda/N) \mathbf{1}_N \mathbf{1}_N^T) \mathbf{e}_i = L \mathbf{e}_i = \lambda_i \mathbf{e}_i, \quad \text{for } i = 2, \dots, N$$

and therefore  $L + (\lambda/N) \mathbf{1}_N \mathbf{1}_N^T$  has the same eigenvalues/eigenvectors of  $L$  for  $i = 2, \dots, N$ . The remaining eigenvalue is associated with the  $\mathbf{e}_1$  eigenvector:

$$(L + (\lambda/N) \mathbf{1}_N \mathbf{1}_N^T) \mathbf{e}_1 = L \mathbf{1}_N + (\lambda) \mathbf{1}_N = \lambda \mathbf{1}_N$$

and its value is  $\lambda$ . As a result, the eigenvalues of  $L + (\lambda/N) \mathbf{1}_N \mathbf{1}_N^T$  are

$$\lambda, \lambda_2(L), \lambda_3(L), \dots, \lambda_N(L).$$

Since we have already that  $\lambda > \bar{\lambda}_2$  (by assumption), and  $\lambda_2(L) \leq \lambda_3(L) \leq \dots \leq \lambda_N(L)$ , the constraint (5) imposes that  $\lambda_2(L) > \bar{\lambda}_2$  and thus it is equivalent to (4).  $\square$

Since for the specified weighted Laplacian  $L(x)$  the maximum value for  $\lambda_2$  is  $N - 1$  [8], we can choose  $\lambda = N$  in (5) and write the maximization of  $\lambda_2$  as

$$\begin{aligned} \mathbf{P}(L(x), \rho_1) : \quad & \max_{x, \gamma} \gamma \\ & \text{s.t. } \gamma > 0 \\ & L(x) + \mathbf{1}_N \mathbf{1}_N^T \succ \gamma I_N \\ & f_d(x_i, x_j) > \rho_1, \quad \forall (i, j) \in \mathcal{E} \end{aligned} \quad (6)$$

The optimal decision variables are the final robot locations  $x$  and the optimal value of  $\gamma$  which is the maximum  $\lambda_2$  for  $L(x)$ . The constraint on  $f_d(x_i, x_j)$  prevents the agents from getting too close to each other and ensures that the trivial solution in which all the agents converge to one point is not part of the feasible solution set of (6).

### 3 Centralized Solution

Problem (6) is non-convex [14] but it is rather standard to obtain a time-varying convex approximation by using first-order Taylor expansions, [11, 14]. Define

$$c_{ij}^w = \frac{\partial f_w}{\partial d_{ij}^2} \frac{\partial d_{ij}^2}{\partial x_i} \Big|_{x_i(k), x_j(k)} = - \frac{\partial f_w}{\partial d_{ij}^2} \frac{\partial d_{ij}^2}{\partial x_j} \Big|_{x_i(k), x_j(k)}, \quad (7)$$

$$[ij] = \frac{\partial f_d}{\partial x_i} \Big|_{x_i(k), x_j(k)} = - \frac{\partial f_d}{\partial x_j} \Big|_{x_i(k), x_j(k)} \quad (8)$$

then

$$w_{ij}(k+1) = w_{ij}(k) + c_{ij}^{w \top} (\delta x_i(k+1) - \delta x_j(k+1)) \quad (9)$$

$$d_{ij}^2(k+1) = d_{ij}^2(k) + [ij]^\top (\delta x_i(k+1) - \delta x_j(k+1)) \quad (10)$$

where  $\delta$  represents the difference operator, i.e.  $\delta x_i(k+1) = x_i(k+1) - x_i(k)$ . The symbol  $\Delta$  will be employed to define the linearized entities; hence the entry  $\Delta \ell_{ij}(x(k+1))$  of the Laplacian  $\Delta L(x(k+1))$  will be

$$\Delta \ell_{ij}(x(k+1)) = \quad (11)$$

$$\begin{cases} 0 & (i, j) \notin \mathcal{E} \\ -w_{ij}(k) - c_{ij}^w{}^\top (\delta x_i(k+1) - \delta x_j(k+1)) & (i, j) \in \mathcal{E}, i \neq j \\ \sum_{l \neq i} w_{il}(k+1) & i = j \end{cases}$$

while

$$\Delta f_d(x_i(k+1), x_j(k+1)) = d_{ij}^2(k) + [i j]^\top (\delta x_i(k+1) - \delta x_j(k+1)) \quad (12)$$

This allows us to consider the maximization of the algebraic connectivity of  $L$  as the following time-varying convex optimization problem [11, 14]:

$$\begin{aligned} \Delta \mathbf{P}(L(x(k)), x(k), \mathcal{S}_{\Delta \mathcal{Q}_2}) : \quad & \max_{x(k+1), \gamma(k+1)} \gamma(k+1) \quad (13) \\ \text{s.t.} \quad & \\ \Delta \mathcal{Q}_1 : \quad & \begin{cases} \gamma(k+1) > 0 \\ \Delta L(x(k+1)) + \mathbf{1}_N \mathbf{1}_N^\top \succ \gamma(k+1) I_N \end{cases} \\ \Delta \mathcal{Q}_2 : \quad & \begin{cases} \mathcal{Q}_{2.1} : \Delta f_d(x_i(k+1), x_j(k+1)) > \rho_1, \\ \quad \quad \quad \forall (i, j) \in \mathcal{E} \\ \mathcal{Q}_{2.2} : \|x_i(k+1) - x_i(k)\| \leq v_{\max} T_s \\ \quad \quad \quad i = 1, \dots, N \end{cases} \end{aligned}$$

where  $\mathcal{S}_{\Delta \mathcal{Q}_2} = \{\rho_1, v_{\max}\}$  represents the parameter set that characterizes the set of constraints  $\Delta \mathcal{Q}_2$ , and it is used to highlight the dependence of the problem on the “physical” limitation of the application scenario (i.e., in this case, the mutual distance  $\rho_1$  and the maximum allowed velocity  $v_{\max}$ ).

In contrast to the original non-convex problem (6), the optimization problem (13) is solved *repeatedly* at each discrete time step  $k$  on-line. In this sense (13) is the  $k$ -th problem of a sequence of convex SDP problems. Note that the achieved maximal algebraic connectivity  $\gamma$  depends on  $k$  and thus we use  $\gamma(k)$ , while the iterative scheme for updating  $\gamma$  is the repeated solution of the optimization problem itself. This means that, letting  $\Delta \mathbf{P}(x(k))$  represent problem (13),  $\gamma$  evolves as

$$(x(k+1), \gamma(k+1)) = \operatorname{argmin} \Delta \mathbf{P}(x(k)).$$

As a consequence of using this sequential convex programming approach (and as a consequence of the non-convex nature of the original problem), although we aim at increasing the cost function at each step  $k$ , we might converge to a local minimum of the original problem (6) and a strong dependence on the initial configuration of the agents has to be expected. Despite these drawbacks,

it has been shown [14] that this formulation does indeed lead to satisfactory local optimal final configurations with a clear increase in the algebraic connectivity.

Assuming that the initial positions  $x(0)$  form a connected graph and the mutual distance between the agents is greater than  $\sqrt{\rho_1}$ , i.e., assuming initial feasibility for the problem, we can prove that the optimization problems (13) will remain feasible for all the subsequent time steps  $k > 0$  (in fact one can always select  $x(k) = x(k+1)$  to obtain a feasible solution) and their solution sequence monotonically increases the algebraic connectivity, [14]. The property of remaining feasible for all  $k$  is related to *persistent* feasibility (also known as *recursive* feasibility), which is a well-known and fundamental concept in the optimization-based control literature [2]. In particular, persistent feasibility ensures that, for any  $k$ , if the  $k$ -th convex problem (13) is feasible then the  $(k+1)$ -st problem will be feasible. This, in addition to initial feasibility (i.e., feasibility at  $k = 0$ ), guarantees that the overall sequential optimization scheme is feasible for all  $k > 0$ . It has to be noted that persistent feasibility ensures only that the solution set of each problem (13) is non-empty, while any improvement in the cost function should be proven separately. However, persistent feasibility is needed to justify the overall optimization scheme in practice.

## 4 More general LTI dynamical models

As our first contribution, we extend the problem (13) in order to allow a more general LTI dynamical model for the agents. Let  $\mathbf{x}_i(\tau) = (x_i(\tau)^\top, v_i(\tau)^\top)^\top$  be the state of agent  $i$  at the discrete time  $\tau$ . We note that the sampling periods belonging to  $\tau$  and  $k$  may differ, meaning that the optimization (13) could be run at a slower rate than the system dynamics. Let the agents have the following second order discrete-time LTI dynamics:

$$\begin{pmatrix} x_i(\tau+1) \\ v_i(\tau+1) \end{pmatrix} = \begin{pmatrix} I_3 & A_{1i} \\ 0_3 & A_{2i} \end{pmatrix} \begin{pmatrix} x_i(\tau) \\ v_i(\tau) \end{pmatrix} + \begin{pmatrix} 0_3 \\ b_{1i} I_3 \end{pmatrix} u_i(\tau) \quad (14)$$

where  $A_{1i} \in \mathbb{R}^{3 \times 3}$ ,  $A_{2i} \in \mathbb{R}^{3 \times 3}$ ,  $b_{1i} \in \mathbb{R}_0$ , and  $u_i(\tau) \in \mathbb{R}^3$  is the control input. Assume:

**Assumption 1** *The matrix  $A_{1i}$  is full rank  $\forall i$ .*

**Assumption 2** *The control input for each agent at each discrete time step is constrained in the closed polytopic set  $\bar{\mathcal{U}}_i$ :*

$$u_i(\tau) \in \bar{\mathcal{U}}_i, \bar{\mathcal{U}}_i = \{u_i(\tau) \in \mathbb{R}^3 | H_i u_i(\tau) \leq h_i\}, \mathbf{0}_3 \in \bar{\mathcal{U}}_i \quad (15)$$

*described via the matrix  $H_i$  and the vector  $h_i$ .*

Assumption 1 is meant to ensure the one-step controllability of the dynamical system described in Eq. (16).

Analogously to  $v_{\max}$  in problem (13), Assumption 2 limits the control input to account for the physical limitations of the agents, and it is a standard formulation of actuator limitations in the optimization-based control community. The state space system in (14) can model agents for which the acceleration does not depend on the position and for which zero velocity and acceleration input ( $v_i(\tau) = 0$  and  $u_i(\tau) = 0$ ) implies  $x_i(\tau + 1) = x_i(\tau)$ . Typically, this class of systems can represent different types of physical agents ranging from fully actuated mobile robots to underwater vehicles. The choice  $A_{1i} = I_3 T_s$ ,  $A_{2i} = I_3$ ,  $b_{1i} = T_s$  yields a double integrator with sampling period  $T_s$ . The reason for the choice of (14) is to consider the simplest model that is capable of showing how to handle the main difficulties when extending the optimization problem (13) to general LTI models. In particular, the key issues are persistent feasibility and collision avoidance. To guarantee persistent feasibility we show how to ensure that  $\mathbf{x}_i(k + 1) = (x_i^\top(k), \mathbf{0}_3^\top)^\top$  is a feasible state for all the agents recalling that the feasibility of the similar solution  $x_i(k + 1) = x_i(k)$  is a sufficient condition for (13) to be persistently feasible. The collision avoidance issue is generated from the fact that the constraint on  $f_d(x_i(k), x_j(k))$  is enforced only at each time step  $k$ , when the optimization problem is solved, but not for every  $\tau$ , which might be a higher rate implementation of the dynamical model. In this respect we show how to ensure that  $f_d(x_i(\tau), x_j(\tau)) > 0$  for every  $\tau$ . We will show that when persistent feasibility and collision avoidance are handled correctly, the problem (13) can be extended to dynamical models of the form (14). In Appendix A we discuss how to possibly cope with these two aspects for an even broader class of LTI dynamical systems.

**Remark 1** *The results of this papers apply to agents modeled via specific LTI dynamical systems. However, the paths (or waypoints) generated for these LTI agents could still be followed by differential drive/tracked vehicles, which see widespread use in mobile robotics. Additional examples include the works of M. M. Zavlanos and co-authors (e.g., [28]) where the discrete-time optimization is used by a continuous-time robot (whose dynamics can be rather arbitrary) in a hybrid systems fashion.*

#### 4.1 Persistent Feasibility

The first step to guarantee persistent feasibility is to ensure that at each time step  $k$  we can affect the position of the agents via the control input. This is not trivial because the position  $x_i(\tau + 1)$  cannot be controlled in one step by  $u_i(\tau)$ . However, we can overcome this issue by solving the optimization problem at a slower rate than the implementation of the control input, e.g., once in two time steps  $\tau$  when we determine both  $u_i(\tau)$  and  $u_i(\tau + 1)$ . In this case the dynamical system (14) can be lifted as seen by the optimization problem:

$$\begin{pmatrix} x_i(\tau + 2) \\ v_i(\tau + 2) \end{pmatrix} = \begin{pmatrix} I_3 & A_{1i}(I_3 + A_{2i}) \\ 0_3 & A_{2i}^2 \end{pmatrix} \begin{pmatrix} x_i(\tau) \\ v_i(\tau) \end{pmatrix} + \begin{pmatrix} b_{1i}A_{1i} & 0_3 \\ b_{1i}A_{2i} & b_{1i}I_3 \end{pmatrix} \begin{pmatrix} u_i(\tau) \\ u_i(\tau + 1) \end{pmatrix} \quad (16)$$

we let  $k = \tau/2$ , and for integer  $k$ 's, we define the lifted variables  $x_i^L(k) = x_i(\tau)$ ,  $v_i^L(k) = v_i(\tau)$ , the lifted state  $\mathbf{x}_i^L(k) = (x_i^L(k)^\top, v_i^L(k)^\top)^\top$ , and the lifted control input  $\mathbf{u}_i^L(k) = (u_i(\tau)^\top, u_i(\tau + 1)^\top)^\top$ . For the sake of simplicity, from now on, we will omit the superscript  $L$  with the idea that if we use the index  $k$  we are referring to the lifted variables. With this in mind, we can rewrite the system (16) using the short-hand notation

$$\mathbf{x}_i(k + 1) = \mathcal{D}_i(\mathbf{x}_i(k), \mathbf{u}_i(k)) \quad (17)$$

We note that the lifted system (17) is controllable to an arbitrary state in one step from  $k$  to  $k + 1$  (see Remark 2 for details). However, the input is constrained to lie in  $\mathbf{u}_i(k) \in \mathcal{U}_i$  (Assumption 2), where  $\mathcal{U}_i = \bar{\mathcal{U}}_i \times \bar{\mathcal{U}}_i$ , i.e.:

$$\mathcal{U}_i = \left\{ \mathbf{u}_i(k) \in \mathbb{R}^6 \left| \begin{pmatrix} H_i \\ H_i \end{pmatrix} \mathbf{u}_i(k) \leq \begin{pmatrix} h_i \\ h_i \end{pmatrix} \right. \right\}, \mathbf{0}_6 \in \mathcal{U}_i \quad (18)$$

Therefore, the next step is to find a feasible control input value  $\mathbf{u}_i(k) \in \mathcal{U}_i$  for which  $\mathcal{D}_i(\mathbf{x}_i(k), \mathbf{u}_i(k)) = (x_i(k)^\top, \mathbf{0}_3^\top)^\top$ . For this reason define the set  $\mathcal{F}_i$  as

$$\begin{aligned} \mathbf{x}_i(k) \in \mathcal{F}_i &\Rightarrow \exists \mathbf{u}_i(k) \in \mathcal{U}_i \text{ such that} \\ \mathcal{D}_i(\mathbf{x}_i(k), \mathbf{u}_i(k)) &= (x_i(k)^\top, \mathbf{0}_3^\top)^\top, \forall k \in \mathbb{N}^+ \end{aligned} \quad (19)$$

For the system (16) the set  $\mathcal{F}_i$  can be computed as the Cartesian product of  $\mathcal{F}_{x,i}$  and  $\mathcal{F}_{v,i}$ , i.e.,  $\mathcal{F}_i = \mathcal{F}_{x,i} \times \mathcal{F}_{v,i}$ , where:

$$\begin{aligned} \mathcal{F}_{x,i} &= \{x_i(k) \in \mathbb{R}^3\}, \quad \text{and} \\ \mathcal{F}_{v,i} &= \end{aligned}$$

$$\left\{ v_i(k) \in \mathbb{R}^3 \left| - \begin{pmatrix} H_i b_{1i}^{-1}(I_3 + A_{2i}) \\ H_i b_{1i}^{-1} A_{2i}(I_3 + 2A_{2i}) \end{pmatrix} v_i(k) \leq \begin{pmatrix} h_i \\ h_i \end{pmatrix} \right. \right\} \quad (20)$$

We note that  $(x_i(k)^\top, \mathbf{0}_3^\top)^\top \in \mathcal{F}_i$ .

#### 4.2 Collision Avoidance

In order to ensure no collisions, a lower bound on  $\rho_1$  has to be determined, which guarantees that if  $f_d(x_i(k), x_j(k)) > \rho_1$  and  $f_d(x_i(k + 1), x_j(k + 1)) > \rho_1$ , then  $f_d(x_i(\tau + 1), x_j(\tau + 1)) > 0$ , for every  $k$  and  $\tau$ . The collision-free condition for any couple  $i$  and  $j$  can be written as

**Remark 2** The dynamical system (16), which is the agent representation seen by the optimization problem, can be written as

$$\mathbf{x}_i(k+1) = \begin{pmatrix} I_3 & A_{1i}(I_3 + A_{2i}) \\ 0_3 & A_{2i}^2 \end{pmatrix} \mathbf{x}_i(k) + \begin{pmatrix} b_{1i}A_{1i} & 0_3 \\ b_{1i}A_{2i} & b_{1i}I_3 \end{pmatrix} \mathbf{u}_i(k) \quad (R1)$$

This system is controllable in one-step by an unconstrained  $\mathbf{u}_i(k)$ . In fact, given an arbitrary state vector  $\mathbf{x}_i(k+1)$  and any initial condition  $\mathbf{x}_i(k)$ , due to the full rank condition on  $A_{1i}$  (Assumption 1), one can promptly invert the system (R1) and obtain the (finite) control vector  $\mathbf{u}_i(k)$ . To see this, consider the dynamical system (R1) and suppose that  $\mathbf{x}_i(k+1)$  and any initial condition  $\mathbf{x}_i(k)$  are given. Then the control input  $\mathbf{u}_i(k)$  can be determined as

$$\begin{aligned} \mathbf{u}_i(k) &= \begin{pmatrix} b_{1i}A_{1i} & 0_3 \\ b_{1i}A_{2i} & b_{1i}I_3 \end{pmatrix}^{-1} \left( \mathbf{x}_i(k+1) - \begin{pmatrix} I_3 & A_{1i}(I_3 + A_{2i}) \\ 0_3 & A_{2i}^2 \end{pmatrix} \mathbf{x}_i(k) \right) \\ &= b_{1i}^{-1} \begin{pmatrix} A_{1i} & 0_3 \\ A_{2i} & I_3 \end{pmatrix}^{-1} \left( \mathbf{x}_i(k+1) - \begin{pmatrix} I_3 & A_{1i}(I_3 + A_{2i}) \\ 0_3 & A_{2i}^2 \end{pmatrix} \mathbf{x}_i(k) \right) \end{aligned}$$

which, is finite by Assumption 1 and  $b_{1i} \in \mathbb{R}_0$ .

$$\begin{aligned} \|x_i(\tau+1) - x_j(\tau+1)\| &\geq \\ \|x_i(\tau) - x_j(\tau)\| - \|A_{1i}v_i(\tau) - A_{1j}v_j(\tau)\| &> 0 \end{aligned} \quad (21)$$

where the triangle inequality is used. Since  $\|x_i(\tau) - x_j(\tau)\| > \sqrt{\rho_1}$  the worst case scenario can be computed maximizing the term  $\|A_{1i}v_i(\tau) - A_{1j}v_j(\tau)\|$  over  $v_i(\tau) \in \mathcal{F}_{v,i}$  and  $v_j(\tau) \in \mathcal{F}_{v,j}$ . This can be rewritten as a non-convex QP problem and pre-solved off-line for any pair  $i$  and  $j$ .<sup>1</sup> If  $\sqrt{\rho_1}$  denotes the worst case  $\|A_{1i}v_i(\tau) - A_{1j}v_j(\tau)\|$  over all the pairs, then the collision-free condition (21) can be expressed as  $\rho_1 > \bar{\rho}_1$ . This is a condition that has to be imposed when designing the  $\rho_1$  value in the minimal distance constraint  $\mathcal{Q}_{2.1}$ . In this respect, we note that the calculations performed to compute  $\bar{\rho}_1$  can be made off-line before running the optimization algorithm (and therefore even the non-convex nature of the problem given the small-size and the off-line calculations can be handled in a satisfactory way in practice).

<sup>1</sup> In order to see this, consider the maximizing of  $\|A_{1i}v_i(\tau) - A_{1j}v_j(\tau)\|$ . This is equivalent to maximize the squared norm  $\|A_{1i}v_i(\tau) - A_{1j}v_j(\tau)\|^2$ , which is equivalent to the following non-convex quadratic program

$$\begin{aligned} \max_{v_i, v_j} \quad & \begin{pmatrix} v_i(\tau) \\ v_j(\tau) \end{pmatrix}^\top \begin{pmatrix} A_{1i}^\top A_{1i} & -A_{1i}^\top A_{1j} \\ -A_{1j}^\top A_{1i} & A_{1j}^\top A_{1j} \end{pmatrix} \begin{pmatrix} v_i(\tau) \\ v_j(\tau) \end{pmatrix} \\ \text{subject to} \quad & v_i(\tau) \in \mathcal{F}_{v,i}, \quad v_j(\tau) \in \mathcal{F}_{v,j} \end{aligned}$$

### 4.3 Optimization Problem

The optimization problem (13) for the maximization of the algebraic connectivity can now be extended for the more general dynamics (14) as

$$\begin{aligned} \Delta \mathbf{P}(\Delta L(x), \mathbf{x}(k), \mathcal{S}_{\Delta \mathcal{Q}_2}) : \quad & \max_{\mathbf{x}(k+1), \mathbf{u}(k), \gamma(k+1)} \gamma(k+1) \\ \text{s.t.} \quad & \\ \Delta \mathcal{Q}_1 : \quad & \begin{cases} \gamma(k+1) > 0 \\ \Delta L(x(k+1)) + \mathbf{1}_N \mathbf{1}_N^\top \succ \gamma(k+1) I_N \end{cases} \\ \Delta \mathcal{Q}_2 : \quad & \begin{cases} \mathcal{Q}_{2.1} : \Delta f_d(x_i(k+1), x_j(k+1)) > \rho_1, \\ \quad \quad \quad \forall (i, j) \in \mathcal{E} \\ \mathcal{Q}_{2.2} : \mathbf{x}_i(k+1) \in \mathcal{F}_i, \quad i = 1, \dots, N \\ \mathcal{Q}_{2.3} : \mathbf{u}_i(k) \in \mathcal{U}_i, \quad i = 1, \dots, N \\ \mathcal{Q}_{2.4} : \mathbf{x}_i(k+1) = \mathcal{D}_i(\mathbf{x}_i(k), \mathbf{u}_i(k)), i = 1, \dots, N \end{cases} \end{aligned} \quad (22)$$

where,  $\mathcal{S}_{\Delta \mathcal{Q}_2} = \{\rho_1, (A_{1i}, A_{2i}, b_{1i}, H_i, h_i)_{i=1, \dots, N}\}$ . As a solution of (22) we find the optimal control inputs  $\mathbf{u}_i(k) = (u_i(\tau)^\top, u_i(\tau+1)^\top)^\top$  that drive the system (14) from  $\mathbf{x}_i(k)$  to  $\mathbf{x}_i(k+1)$ . We define the concept of feasible state as follows.

**Definition 1** A state  $x(k)$  is feasible if  $\mathbf{x}_i(k) \in \mathcal{F}_i, \forall i$ ,  $\Delta L(x(k)) + \mathbf{1}_N \mathbf{1}_N^\top \succ 0$ , and  $d_{ij}^2(k) > \rho_1 \forall (i, j) \in \mathcal{E}$ .

For the optimization problem (22), as in [14], we assume initial feasibility for the first time instance:

**Assumption 3** The initial state  $\mathbf{x}(0)$  is a feasible state.

The following theorem states formally the persistent feasibility property:

**Theorem 1** *If for any discrete time  $k$ ,  $\mathbf{x}(k)$  is a feasible state according to Definition 1, then the problem (22) will be feasible for the discrete time  $k + 1$ .*

*Proof.* Consider  $\mathbf{x}_i(k+1) = (x_i(k)^\top, \mathbf{0}_3^\top)^\top$  as the solution of the optimization (22) at time  $k+1$ . This solution satisfies  $\mathcal{Q}_{2.1}$ ,  $\mathcal{Q}_{2.2}$ , and  $\mathcal{Q}_{2.3}$ . Moreover, since  $\mathbf{x}_i(k) \in \mathcal{F}_i$  by assumption, there exist control inputs  $\mathbf{u}_i(k) \in \mathcal{U}_i$  for all the agents for which  $(x_i(k)^\top, \mathbf{0}_3^\top)^\top = \mathcal{D}_i(\mathbf{x}_i(k), \mathbf{u}_i(k))$ . Therefore the solution  $\mathbf{x}_i(k+1)$  satisfies  $\mathcal{Q}_{2.3}$  and  $\mathcal{Q}_{2.4}$  and thus the claim.  $\square$

Combining Theorem 1 with Assumption 3, it follows that the sequence of problems (22) is feasible for all  $k > 0$ . We note that persistent feasibility (Theorem 1) is a fundamental property to guarantee that the overall optimization scheme remains feasible, while we show later (in the distributed case) that the sequence of solutions lead to a monotonic increase of the cost function.

The reasons for the initial choices of  $k = \tau/2$  and  $\mathcal{F}_i$  should be clearer after Theorem 1. The fact that  $\mathbf{x}_i(k) \in \mathcal{F}_i$  guarantees that the solution  $\mathbf{x}_i(k+1) = (x_i(k)^\top, \mathbf{0}_3^\top)^\top$  is feasible in terms of admissible control action, which is a sufficient condition to guarantee that the optimization problem (22) is persistently feasible. The choice  $k = \tau/2$  ensures that  $\mathcal{F}_i$  is always non-empty.

## 5 Distributed Solution

In this section we present our main contribution: a non-iterative and guaranteed feasible *distributed* solution to solve (22). We note that this is not a trivial task, since commonly used decomposition methods for optimization problems (if applicable, e.g. in [9]) typically require iterative solutions which may not be amenable to fast real-time implementations.

Our solution depends on subproblems which each agent solves locally and whose size can be decided according to the available resources. This size is influenced by the notion of an enlarged neighborhood set, collecting all the agents whose data are available locally at each time step  $k$ . The proposed distributed solution is computed in two phases. The first step is to solve a local optimization problem that is a small-scale *modified* version of the centralized problem, in which the farthest agents (in terms of graph distance, i.e. minimum number of connecting edges) are constrained to be stationary, i.e.  $\mathbf{x}_i(k+1) = (x_i(k)^\top, \mathbf{0}_3^\top)^\top$ . This step is similar to a Jacobi-type optimization [1], where only certain variables are updated at a time, but also differs in the modification of the local problems and their reduced size. The second step is to share the proposed solutions within the enlarged neighborhood and combine them using an agent-dependent positive linear combination. We note that this sharing/combining procedure is performed just once for each optimization step, making

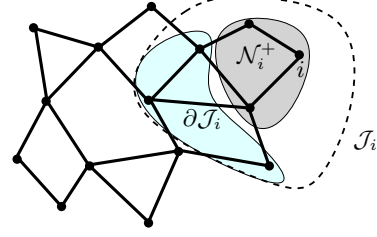


Fig. 2. Notation for the distributed solution in case the size of the enlarged neighborhood size for agent  $i$  is  $n_i = 2$ . The thick lines represent links between connected agents.

the overall scheme non-iterative in contrast with commonly used consensus algorithms. The key point in the proposed distributed solution is to *jointly* construct the feasible local problems with modified local constraints *and* the positive linear combination of the solutions to preserve feasibility of the global solution and a monotonically increasing cost function.

The most important novelty of the proposed solution is the idea of modifying the local problems and designing the merging mechanism to ensure feasibility and improvement properties for the locally merged solution with respect to the original centralized problem. This idea, as remarked in the Introduction, offers a complementary approach to standard subgradient algorithms [1], which can be thought of as being on the opposite side of the “communication-computation” trade-off spectrum.

Let  $\mathcal{J}_i$  denote the enlarged neighborhood of  $i$  consisting of all the agents whose state is known by agent  $i$  at each sampling time  $k$  (either through direct or indirect communication). We define this set in a recursive way: let  $\mathcal{N}_i^1$  be the standard, first-order neighborhood of  $i$ , i.e.  $\mathcal{N}_i^1 = \mathcal{N}_i^+$ , then, the  $n_i$ -size enlarged neighborhood of  $i$  for  $n_i > 1$  is defined as

$$\mathcal{J}_i = \mathcal{N}_i^{n_i} = \bigcup_{j \in \mathcal{N}_i^{n_i-1}} \mathcal{N}_j^{n_i-1} \quad (23)$$

in other words, the collection of the  $(n_i - 1)$ -size enlarged neighborhoods of all  $j \in \mathcal{N}_i^{n_i-1}$ . The scalar  $n_i \geq 1$  implies bounds on the diameter of the communication graph constructed with the agents in  $\mathcal{J}_i$ . The cardinality of  $\mathcal{J}_i$  is  $J_i$ . We call the set of agents belonging to  $\partial \mathcal{J}_i$ , the bordering agents of  $\mathcal{J}_i$  defined as

$$\partial \mathcal{J}_i = \{j | j \in \mathcal{J}_i, j \notin \mathcal{N}_i^{n_i-1}\} \quad (24)$$

Denote the graph Laplacian associated with the communication graph corresponding to the agents in  $\mathcal{J}_i$  as  $L_{i,n_i}$  and the communication link set as  $\mathcal{E}_{i,n_i}$ . Figure 2 provides a graphical illustration of this notation for  $n_i = 2$ . Define  $\mathbf{x}_{\mathcal{J}_i}$  and  $\mathbf{u}_{\mathcal{J}_i}$  as the stacked vectors collecting the states and the lifted control inputs for all the agents  $j$  belonging to the enlarged neighborhood of  $i$ , i.e.  $j \in \mathcal{J}_i$ .

As a first step of the distributed solution, for each agent  $i$ , we consider local problems  $\Delta \mathbf{P}_i$  of the form:

$$\begin{aligned} \Delta \mathbf{P}_i(\Delta L_{i,n_i}(x_{\mathcal{J}_i}), \mathbf{x}_{\mathcal{J}_i}(k), \mathcal{S}_{\Delta \hat{\mathcal{Q}}_{2i}}) : \quad (25) \\ \max_{\mathbf{x}_{\mathcal{J}_i}(k+1), \mathbf{u}_{\mathcal{J}_i}(k), \gamma_i(k+1)} \gamma_i(k+1) \\ \text{s.t.} \\ \Delta \mathcal{Q}_1 : \begin{cases} \gamma_i(k+1) > 0 \\ \Delta L_{i,n_i}(x_{\mathcal{J}_i}(k+1)) + \mathbf{1}_{J_i} \mathbf{1}_{J_i}^T \succ \gamma_i(k+1) I_{J_i} \end{cases} \\ \Delta \hat{\mathcal{Q}}_{2i} : \begin{cases} \hat{\mathcal{Q}}_{2.1} : \Delta f_d(x_i(k+1), x_j(k+1)) > \hat{\rho}_{1ij}, \\ \quad \forall (i, j) \in \mathcal{E}_{i,n_i} \\ \hat{\mathcal{Q}}_{2.2} : \mathbf{x}_j(k+1) \in \hat{\mathcal{F}}_j = \mathcal{F}_j, \quad j \in \mathcal{J}_i \\ \hat{\mathcal{Q}}_{2.3} : \mathbf{u}_j(k) \in \hat{\mathcal{U}}_j, \quad j \in \mathcal{J}_i \\ \hat{\mathcal{Q}}_{2.4} : \mathbf{x}_j(k+1) = \hat{\mathcal{D}}_j(\mathbf{x}_j(k), \mathbf{u}_j(k)), \quad j \in \mathcal{J}_i \end{cases} \\ \mathcal{Q}_3 : \mathbf{x}_j(k+1) = (x_j(k)^\top, \mathbf{0}_3^\top)^\top, \quad \text{for } j \in \partial \mathcal{J}_i \end{aligned}$$

Where  $\mathcal{S}_{\Delta \hat{\mathcal{Q}}_{2i}} = \{(\hat{\rho}_{1ij}, \hat{A}_{1j}, \hat{A}_{2j}, \hat{b}_{1j}, \hat{H}_j, \hat{h}_j)_{j \in \mathcal{J}_i}\}$  and the notation  $\hat{\mathcal{D}}_j$  denotes a dynamical system of the same form as (16) but with the modified triplet  $(\hat{A}_{1j}, \hat{A}_{2j}, \hat{b}_{1j})$ . We will show later (Theorem 2) how to construct the modified state matrices and parameter set  $\mathcal{S}_{\Delta \hat{\mathcal{Q}}_{2i}}$ .

The optimal local decision variables (solution of  $\Delta \mathbf{P}_i$ ) will be denoted as  $\hat{\gamma}_i(k+1)$ ,  $\hat{\mathbf{x}}_{\mathcal{J}_i}(k+1)$ , and  $\hat{\mathbf{u}}_{\mathcal{J}_i}(k)$  respectively. We call  $\hat{\mathbf{x}}_{ij}(k+1)$  the state of agent  $j$  as computed by agent  $i$  and we use the same notation for  $\hat{\mathbf{u}}_{ij}(k)$ . We note that the optimal local decision variables  $\hat{\mathbf{x}}_{\mathcal{J}_i}(k+1)$  and  $\hat{\mathbf{u}}_{\mathcal{J}_i}(k)$  are composed of  $\hat{\mathbf{x}}_{ij}(k+1)$  and  $\hat{\mathbf{u}}_{ij}(k)$  for each  $j \in \mathcal{J}_i$ . We emphasize that the extra constraint  $\mathcal{Q}_3$  is an important requirement to guarantee feasibility, as will be explained shortly in this section. We will also require  $\hat{\mathcal{F}}_i = \mathcal{F}_i$  for all the agents as a sufficient condition of persistent feasibility.

Consider the set of all agents  $p$  which have agent  $i$  inside their local problems  $\Delta \mathbf{P}_p$ , i.e.  $i \in \mathcal{J}_p$ , and denote by  $\mathcal{J}_i^* = \{p | i \in \mathcal{J}_p\}$ . Since the enlarged neighborhood size  $n_i$  could differ from agent to agent,  $\mathcal{J}_i^* \neq \mathcal{J}_i$ .

As a second step of the distributed solution, we construct the position update based on the previous solution  $\mathbf{x}(k)$  and a positive linear combination of the local position solutions  $\hat{x}_{\mathcal{J}_i}(k)$  as:

$$x_i(k+1) = x_i(k) + \sum_{j \in \mathcal{J}_i^*} \alpha_j \delta \hat{x}_{ji}(k+1), \quad i = 1, \dots, N \quad (26)$$

for  $\alpha_j > 0$  (recall that, since  $\hat{x}_{ij}(k) = x_i(k)$ ,  $\delta \hat{x}_{ji}(k+1) = \hat{x}_{ji}(k+1) - x_i(k)$ ). Define

$$\bar{\alpha}_i = \sum_{j \in \mathcal{J}_i^*} \alpha_j$$

and observe that  $\bar{\alpha}_i$  is in general not equal to one. We require  $\bar{\alpha}_i \leq 1$  due to the linearization procedure, in

fact, bigger  $\bar{\alpha}_i$  would question the validity of the Taylor expansions in the local problems.

We prove the following lemma regarding the sum of local position solutions, which is instrumental for the subsequent theorems.

**Lemma 1** For arbitrary vectors  $q_{ij} \in \mathbb{R}^3$  where  $(i, j)$  are neighbors (i.e. if  $\ell_{ij} \neq 0$ ), and for any  $\delta \hat{x}_{pi}(k+1), \delta \hat{x}_{pj}(k+1)$  part of the optimal solutions of the local problems  $\Delta \mathbf{P}_p$  in (25), with  $p \in \mathcal{J}_i^*$  and  $p \in \mathcal{J}_j^*$  respectively, the following equality holds:

$$\begin{aligned} q_{ij}^\top \left( \sum_{p \in \mathcal{J}_i^*} \alpha_p \delta \hat{x}_{pi}(k+1) - \sum_{p \in \mathcal{J}_j^*} \alpha_p \delta \hat{x}_{pj}(k+1) \right) = \\ q_{ij}^\top \sum_{p \in \mathcal{J}_i^* \cap \mathcal{J}_j^*} (\delta \hat{x}_{pi}(k+1) - \delta \hat{x}_{pj}(k+1)) \quad (27) \end{aligned}$$

*Proof.* The first term of the equality (27) can be divided into three parts:  $p \in \mathcal{J}_i^* \cap \mathcal{J}_j^*$ ,  $p \in \mathcal{J}_i^* \wedge p \notin \mathcal{J}_j^*$ , and  $p \in \mathcal{J}_j^* \wedge p \notin \mathcal{J}_i^*$ . Since we are interested in the case when  $i$  and  $j$  are neighbors, we can make the key observations that:

$$\{p | p \in \mathcal{J}_i^* \wedge p \notin \mathcal{J}_j^*\} \Rightarrow i \in \partial \mathcal{J}_p \quad (28)$$

$$\{p | p \in \mathcal{J}_j^* \wedge p \notin \mathcal{J}_i^*\} \Rightarrow j \in \partial \mathcal{J}_p \quad (29)$$

Consider the first implication (28). If  $p \in \mathcal{J}_i^*$ , then  $i$  and  $p$  are separated by at most  $n_p$  links. Furthermore, if  $p \notin \mathcal{J}_j^*$ , then  $j$  and  $p$  are separated by at least  $n_p + 1$  links. Since  $i$  and  $j$  are neighbors, it follows that the separation between  $i$  and  $p$  is exactly  $n_p$  links and therefore  $i \in \partial \mathcal{J}_p$ . The second implication (29) can be proven by similar arguments. The two implications (28)-(29) allow us to rewrite the first part of the equality (27) as:

$$\begin{aligned} q_{ij}^\top \sum_{p \in \mathcal{J}_i^* \cap \mathcal{J}_j^*} \alpha_p (\delta \hat{x}_{pi}(k+1) - \delta \hat{x}_{pj}(k+1)) + \\ \underbrace{q_{ij}^\top \sum_{p \in \mathcal{J}_i^* \wedge p \notin \mathcal{J}_j^*} \alpha_p \delta \hat{x}_{pi}(k+1)}_{=0} - \underbrace{q_{ij}^\top \sum_{p \in \mathcal{J}_j^* \wedge p \notin \mathcal{J}_i^*} \alpha_p \delta \hat{x}_{pj}(k+1)}_{=0} \end{aligned}$$

where the last two terms are 0 due to (28)-(29) and the constraint  $\mathcal{Q}_3$  of  $\Delta \mathbf{P}_p$  in (25), which requires  $\delta \hat{x}_{pi}(k+1) = 0$  and  $\delta \hat{x}_{pj}(k+1) = 0$  for  $i \in \partial \mathcal{J}_p$  and  $j \in \partial \mathcal{J}_p$ , respectively.  $\square$

We are ready to construct the parameter set  $\mathcal{S}_{\Delta \hat{\mathcal{Q}}_{2i}}$  which defines the local set of constraints  $\Delta \hat{\mathcal{Q}}_{2i}$ .

**Theorem 2** (Local constraints for global feasibility) Taking for each  $i$ , the following choices:



- the local parameter set  $\Delta\hat{\mathcal{Q}}_{2i}$  in (25) as

$$\mathcal{S}_{\Delta\hat{\mathcal{Q}}_{2i}} = \{(\hat{\rho}_{1ij}, \bar{\alpha}_j^{-1}A_{1j}, A_{2j}, \bar{\alpha}_jb_{1j}, H_j, \bar{\alpha}_j^{-1}h_j)_{j \in \mathcal{J}_i}\}$$

meaning:  $\tilde{A}_{1j} = \bar{\alpha}_j^{-1}A_{1j}$ ,  $\tilde{A}_{2j} = A_{2j}$ ,  $\tilde{b}_{1j} = \bar{\alpha}_jb_{1j}$ ,  $\tilde{H}_j = H_j$ ,  $\tilde{h}_j = \bar{\alpha}_j^{-1}h_j$ , and

$$\hat{\rho}_{1ij} = \bar{\alpha}_{ij}^{-1}(\rho_1 + d_{ij}^2(k)(\bar{\alpha}_{ij} - 1)) \quad (30)$$

with  $\bar{\alpha}_{ij} = \sum_{p \in \mathcal{J}_i^* \cap \mathcal{J}_j^*} \alpha_p$ ;

- the positive linear combination of the local optimal control inputs  $\hat{\mathbf{u}}_{ji}(k)$  in (25) as

$$\mathbf{u}_i(k) = \sum_{j \in \mathcal{J}_i^*} \alpha_j \hat{\mathbf{u}}_{ji}(k) \quad (31)$$

- the positive linear combination of the local optimal velocities  $\hat{v}_{ji}(k+1)$  in (25) as

$$v_i(k+1) = \frac{\sum_{j \in \mathcal{J}_i^*} \alpha_j \hat{v}_{ji}(k+1)}{\bar{\alpha}_i} \quad (32)$$

ensure that the updated position vector  $x(k+1)$ , the control vector  $\mathbf{u}(k)$ , and velocity vector  $v(k+1)$  based on (26), (31), and (32) respectively, satisfy the set of constraints  $\Delta\mathcal{Q}_2$  of the global problem (22).

*Proof.* We give a constructive proof of the theorem in Appendix B.

Theorem 2 not only gives a procedure to construct the local constraints so that the linear combination (26) satisfies the global constraints, it also establishes a link between the local quantities and the global ones. Furthermore, it ensures that in order to move to the updated state  $\mathbf{x}_i(k+1)$  (comprised of the position update  $x_i(k+1)$  and the velocity update  $v_i(k+1)$ ) each agent can implement the linear combination of the lifted control input (31) as summarized in Algorithm 1, without explicitly computing the merging (26) or (32). In this context, from a control perspective, each local optimization problem (25) computes the control input  $\hat{\mathbf{u}}_{ji}(k)$ . This is merged via (31) with the enlarged neighborhood information and then implemented. This merged control by Theorem 2 induces the merging mechanisms on the state (26) and (32), which therefore do not have to be computed explicitly.

## 6 Properties of the Distributed Solution

In the previous section we have seen how to construct the local problem parameter set  $\mathcal{S}_{\Delta\hat{\mathcal{Q}}_{2i}}$  and positive linear combinations of the local solutions to ensure that the combined solution  $(\mathbf{x}(k+1), \mathbf{u}(k))$  satisfies the constraint  $\Delta\mathcal{Q}_2$  of the global problem (13). In this section

---

### Algorithm 1 Distributed $\lambda_2$ Maximization.

---

1. Input for each agent  $i$ :  $\mathbf{x}_j(k)$ ,  $j \in \mathcal{J}_i$
2. Solve:  $\Delta\mathbf{P}_i$  in (23) computing

$$(\hat{\mathbf{x}}_{ji}(k+1), \hat{\mathbf{u}}_{ji}(k+1)), j \in \mathcal{J}_i$$

3. Communicate:  $\hat{\mathbf{u}}_{ji}(k+1)$  among members of  $\mathcal{J}_i$
4. Positive linear combination:

$$\mathbf{u}_i(k) = \sum_{j \in \mathcal{J}_i^*} \alpha_j \hat{\mathbf{u}}_{ji}(k)$$

5. Implement the control action  $\mathbf{u}_i(k)$
- 

we will look at  $\Delta\mathcal{Q}_1$  and at the persistent feasibility of Algorithm 1. Theorem 3 and 4 will establish that

**(C1)** The algebraic connectivity of the global linearized Laplacian  $\Delta L(x(k+1))$  of (13) with  $x(k+1)$  computed via (26) is monotonically increasing in each iteration, which implies that  $x(k+1)$  will also satisfy  $\Delta\mathcal{Q}_1$  of the global problem (13) for a certain value of  $\gamma(k+1) \geq \gamma(k)$ .

This is proven linking the linear combination (26) and the algebraic connectivity through the linear dependence of the linearized Laplacian on the position  $x$ . Theorem 5 will show that

**(C2)** The distributed optimization in Algorithm 1 is persistently feasible using the constructed  $\Delta\hat{\mathcal{Q}}_{2i}$ 's in Theorem 2.

This is proven by the use of the relation between local and global feasibility of Theorem 2.

First of all, reconsider the linearized Laplacian  $\Delta L(x(k+1))$  entries, given in (??). We can rewrite  $\Delta L(x(k+1))$  as a sum

$$\Delta L(x(k+1)) = \Delta L(\delta x(k+1)) + L(x(k)).$$

Under the validity of the employed Taylor approximation, we assume that for all practical situations the value of  $L(x(k))$  is equivalent to its linearized approximation  $\Delta L(x(k))$ , and therefore we can write

$$\Delta L(x(k+1)) = \Delta L(\delta x(k+1)) + \Delta L(x(k)) = L(x(k+1)). \quad (33)$$

Consider the local problem  $\Delta\mathbf{P}_i$  in (25), and its solution comprised of  $\hat{x}_{ij}(k+1)$  for all  $j \in \mathcal{J}_i$ . Construct the global vector  $\hat{x}^{(i)}(k+1)$  whose entries are determined based on the local solution as

$$\begin{aligned} \hat{x}^{(i)}(k+1) &= (\dots, \hat{x}_j^{(i)}(k+1)^\top, \dots)^\top, \quad j = 1, \dots, N \\ \text{with } \hat{x}_j^{(i)}(k+1) &= \begin{cases} \hat{x}_{ij}(k+1) & \text{if } j \in \mathcal{J}_i \\ x_j(k) & \text{otherwise} \end{cases} \end{aligned} \quad (34)$$

where we keep those agent positions that have not been optimized fixed, and we update the rest from the solution of the local problem.

**Theorem 3** (C1.a) *The positions  $\hat{x}^{(i)}(k+1)$  in (34) constructed from the solution of the local problem  $\Delta\mathbf{P}_i$  in (25), monotonically increase the algebraic connectivity of the Laplacian matrix:*

$$\Delta L(\tilde{x}^{(i)}(k+1)) \succeq \Delta L(x(k)). \quad (35)$$

*Proof.* Since  $\Delta L$  depends linearly on the position  $x$  by (33) we can write

$$\Delta L(\tilde{x}^{(i)}(k+1)) = \Delta L(\delta\tilde{x}^{(i)}(k+1)) + \Delta L(x(k)),$$

thus the relation (35) can be interpreted as

$$\Delta L(\delta\tilde{x}^{(i)}(k+1)) \succeq 0. \quad (36)$$

We recall that,

*First:* for (34)  $\delta\tilde{x}_j^{(i)}(k+1) = 0$  if  $j \notin \mathcal{J}_i$ .

*Second:* for the constraint  $\mathcal{Q}_3$  in the local problem  $\Delta\mathbf{P}_i$  (25),  $\delta\tilde{x}_j^{(i)}(k+1) = 0$  if  $j \in \partial\mathcal{J}_i$ .

For these two observations,  $[\Delta L(\delta\tilde{x}^{(i)}(k+1))]_{ij} \neq 0$  only if  $(i, j) \in \mathcal{E}_{i, n_i}$  and therefore up to a reordering the Laplacian  $\Delta L(\delta\tilde{x}^{(i)}(k+1))$  has the form

$$\left[ \begin{array}{c|c} \Delta L_{i, n_i}(\delta\tilde{x}_{\mathcal{J}_i}(k+1)) & 0 \\ \hline 0 & 0 \end{array} \right] \succeq 0. \quad (37)$$

We recall that  $\tilde{x}_{\mathcal{J}_i}(k+1)$  is the optimal decision variable for the position in the local optimization problems (and the order of the single elements is not important).

We can now restate (36) via (37) as

$$\Delta L_{i, n_i}(\delta\tilde{x}_{\mathcal{J}_i}(k+1)) \succeq 0$$

or

$$\Delta L_{i, n_i}(\tilde{x}_{\mathcal{J}_i}(k+1)) \succeq \Delta L_{i, n_i}(\tilde{x}_{\mathcal{J}_i}(k))$$

which is true due to the local optimality of the local solution of  $\Delta\mathbf{P}_i$ .  $\square$

We can relate the positions  $\hat{x}^{(i)}(k+1)$  in (34) with  $x_i(k+1)$  in (26), by the following Lemma.

**Lemma 2** *When considering the positions  $\hat{x}^{(i)}(k+1)$  in (34) and  $x_i(k+1)$  in (26) the following equality holds:*

$$\Delta L(\delta x(k+1)) = \sum_{i=1}^N \alpha_i \Delta L(\delta\hat{x}^{(i)}(k+1)) \quad (38)$$

*Proof.* Let us consider the entry  $(i, j)$  of the Laplacian  $\Delta L$  on both sides of the expression. For the right side,  $\ell_{ij}^{\text{right}}$  can be expressed as

$$\ell_{ij}^{\text{right}} = c_{ij}^{w^\top} \sum_{p \in \mathcal{J}_i^* \cap \mathcal{J}_j^*} \alpha_p (\delta\hat{x}_{pi}(k+1) - \delta\hat{x}_{pj}(k+1))$$

since the entry  $(i, j)$  will exist only for the subproblems  $\Delta\mathbf{P}_p$  with  $p \in \mathcal{J}_i^* \cap \mathcal{J}_j^*$ . For the left side,

$$\begin{aligned} \ell_{ij}^{\text{left}} &= c_{ij}^{w^\top} (\delta x_i(k+1) - \delta x_j(k+1)) = \\ &= c_{ij}^{w^\top} \left( \sum_{p \in \mathcal{J}_i^*} \alpha_p \delta\hat{x}_{pi}(k+1) - \sum_{p \in \mathcal{J}_j^*} \alpha_p \delta\hat{x}_{pj}(k+1) \right) \end{aligned}$$

The coefficient  $c_{ij}^{w^\top}$  is non-zero only if  $(i, j)$  are neighbors and using Lemma 1 leads to

$$\ell_{ij}^{\text{left}} = c_{ij}^{w^\top} \sum_{p \in \mathcal{J}_i^* \cap \mathcal{J}_j^*} \alpha_p (\delta\hat{x}_{pi}(k+1) - \delta\hat{x}_{pj}(k+1)) \quad \square$$

Using Theorem 3 and Lemma 2 we can now prove the monotonically increasing property of the algebraic connectivity of the global linearized Laplacian  $\Delta L(x(k+1))$ .

**Theorem 4** (C1.b) *The algebraic connectivity of the global linearized Laplacian  $\Delta L(x(k+1))$  is monotonically increasing in each iteration, meaning  $\Delta L(x(k+1)) \succeq \Delta L(x(k))$ , where  $x(k+1)$  is computed by the combination (26).*

*Proof.* Theorem 3 implies  $\Delta L(\delta\tilde{x}^{(i)}(k+1)) \succeq 0$  for all  $i$ . Thus summing over all agents leads to

$$\sum_{i=1}^N \alpha_i \Delta L(\delta\tilde{x}^{(i)}(k+1)) \succeq 0$$

Considering the linear combination  $x_i(k+1)$  in (26), and the associated global vector  $x(k+1)$ , by Lemma 2 it follows that  $\Delta L(\delta x(k+1)) \succeq 0$ . From the linear dependence of  $\Delta L$  on  $x$  (Equation (33)),

$$\Delta L(x(k+1)) = \Delta L(\delta x(k+1)) + \Delta L(x(k))$$

and therefore it follows that  $\Delta L(x(k+1)) - \Delta L(x(k)) \succeq 0$  and the desired property:  $\Delta L(x(k+1)) \succeq \Delta L(x(k))$ .  $\square$

Finally, we can show the persistent feasibility of the distributed optimization algorithm (Algorithm 1).

**Theorem 5 (C2)** *The distributed optimization algorithm presented in Algorithm 1 is persistently feasible.*

*Proof.* We have to prove that if, for any discrete time  $k$ ,  $\mathbf{x}(k)$  is a feasible initial state for the global optimization problem  $\Delta\mathbf{P}$  (22) at the discrete time  $k$  (Definition 1), then there will be a feasible solution to the distributed optimization problem in Algorithm 1. Such a feasible solution can be thought of as an initial state  $\mathbf{x}(k+1)$  for the global optimization problem  $\Delta\mathbf{P}$  (22) at the discrete time  $k+1$ . We prove the existence of such feasible solution in two steps.

*Step 1.* Using the assumption that  $\mathbf{x}(k)$  is a feasible initial state for the global optimization problem  $\Delta\mathbf{P}$  (22) at time step  $k$ , we can show that  $\mathbf{x}(k)$  is also a feasible initial state for the local problems  $\Delta\mathbf{P}_i$  (25), which therefore are feasible and deliver local solutions  $(\hat{\mathbf{x}}_{\mathcal{J}_i}(k+1), \hat{\mathbf{u}}_{\mathcal{J}_i}(k+1))$  satisfying the constraints  $\Delta\mathcal{Q}_1$ ,  $\Delta\hat{\mathcal{Q}}_{2i}$ , and  $\mathcal{Q}_3$ . This claim follows from Theorem 2, in particular from the fact that  $\hat{\rho}_{1ij} \leq \rho_1$ . In fact, from the assumption  $\bar{\alpha} \leq 1$  and  $d_{ij}^2(k) > \rho_1$  (feasibility at  $k$ ), the relation (30) yields  $\hat{\rho}_{1ij} \leq \rho_1$ , and thus  $\mathbf{x}(k)$  is also a feasible initial state for the local problems  $\Delta\mathbf{P}_i$  (25).

*Step 2.* We can show that after merging/combining the resulting local solutions  $(\hat{\mathbf{x}}_{\mathcal{J}_i}(k+1), \hat{\mathbf{u}}_{\mathcal{J}_i}(k+1))$ , the final distributed state solution  $\mathbf{x}(k+1)$  will be a feasible initial state for the global optimization problem  $\Delta\mathbf{P}$  in (22) at the discrete time  $k+1$ . This second step follows directly from Theorem 2 and Theorem 4.  $\square$

Similarly to Theorems 2 and 4, we note that Theorem 5 holds also if the agents change the size of their enlarged neighborhood  $n_i$  from time step  $k$  to  $k+1$ , since the feasibility of the state in the local problems does not depend on the enlarged neighborhood size of  $\mathcal{J}_i$ . This fact will be used in the next section to allow adjusting the communication load of each agent and make Algorithm 1 adaptive.

## 7 Adapting the Communication Load

In this section we investigate further the properties of the distributed solution presented in Section 5. First we show in Theorem 6 that if all-to-all communication is allowed then the distributed solution of Algorithm 1 is equivalent<sup>2</sup> to the centralized approach in (22). Then we prove in Theorem 7 that starting from the same state vector  $\mathbf{x}(k)$ , if we run Algorithm 1 with different enlarged neighborhood sizes, the solution that delivers a higher algebraic connectivity at time step  $k+1$  is the one with the greater neighborhood size  $n$ . This last fact enables us

<sup>2</sup> Meaning that the two solutions (centralized and distributed) are the same.

to characterize a local relative sub-optimality measure with respect to a larger enlarged neighborhood size.

**Theorem 6 (Equivalence)** *The distributed solution of Algorithm 1 is equivalent to the centralized one of (22), if all-to-all communication is allowed (meaning  $n_i = N$ ,  $\forall i$ , and thus no bordering agents) and if  $\alpha_i = 1/N$ ,  $\forall i$ , is chosen as weight in the positive linear combinations of the local states and inputs (26), (32), and (31).*

*Proof.* Consider  $n_i = N$ ,  $\partial\mathcal{J}_i = \{\emptyset\}$  for all the agents, and the choice  $\alpha_i = 1/N$ ,  $\forall i$ . We have  $\bar{\alpha}_i = 1$  and  $\sum_{p \in \mathcal{J}_i^* \cap \mathcal{J}_i^*} \alpha_p = 1$ . Therefore, as a consequence of the choices of Theorem 2,  $\Delta\hat{\mathcal{Q}}_{2i} \equiv \Delta\mathcal{Q}_2$ . Furthermore, all the constructed local solutions  $\hat{x}^{(i)}(k+1)$  in (34) are the same and they are equivalent to the solution of the centralized problem  $x(k+1)$  in (22). Given the specified selection of  $\alpha_i$ , also the linear combination (26) is equivalent to  $\hat{x}^{(i)}(k)$  and therefore the distributed position solution delivered by Algorithm 1 is equivalent to the centralized one of (22). Since the same arguments hold for the control inputs and velocities the claim is proven.  $\square$

**Definition 2** *The vector  $x^{(i)}(k+1)|_{n_i}$  is the constructed local solution (34) using an enlarged neighborhood size  $n_i$  in the local problem  $\Delta\mathbf{P}_i$  (25).*

**Definition 3** *The vector  $x(k+1)|_{\mathbf{n}}$  is the global solution of Algorithm 1 at step  $k+1$ , with  $\mathbf{n} = (n_1, \dots, n_N)$ .*

Using the above definitions, we can prove the following theorem about the effect of an increased neighborhood size on the resulting algebraic connectivity.

**Theorem 7** *If  $\mathbf{n}_1 \geq \mathbf{n}_2$  element-wise, then the algebraic connectivity of  $\Delta L(x(k+1)|_{\mathbf{n}_2})$  is greater than equal to the one of  $\Delta L(x(k+1)|_{\mathbf{n}_1})$ , implying  $\Delta L(x(k+1)|_{\mathbf{n}_2}) \succeq \Delta L(x(k+1)|_{\mathbf{n}_1})$ .*

*Proof.* By optimality and due to the linearity of  $L$  on  $x$  (Eq. (33)), for each  $i$  we can state

$$\Delta L\left(\delta x^{(i)}(k+1)|_{\mathbf{n}_{2,i}}\right) \succeq \Delta L\left(\delta x^{(i)}(k+1)|_{\mathbf{n}_{1,i}}\right)$$

Multiplying by  $\alpha_i$  and summing over  $i$  leads to

$$\sum_{i=1}^N \alpha_i \Delta L\left(\delta x^{(i)}(k+1)|_{\mathbf{n}_{2,i}}\right) \succeq \sum_{i=1}^N \alpha_i \Delta L\left(\delta x^{(i)}(k+1)|_{\mathbf{n}_{1,i}}\right)$$

By Lemma 2 the claim follows.  $\square$

We note that Theorem 7 holds considering one step horizon, i.e., from  $k$  to  $k+1$ . Due to the non-linear/non-convex nature of the original problem (6), this result

does not hold in general from  $k$  to  $k + 2$  or beyond, as we will show in the simulation experiments of Section 8.

Theorem 7 is instrumental to construct a measure that can be used to decide locally on-line whether to increase or decrease the size  $n_i$  of the enlarged neighborhood. This measure can be used to adapt  $n_i$  to influence the trade-off between the increase of the algebraic connectivity or the reduction of the communication cost. For this purpose, we define two local relative sub-optimality measures with respect to a larger enlarged neighborhood size as

$$e_i^+ = 1 - \frac{\lambda_2(\Delta L_{i,n_i+1}(x^{(i)}(k+1)|_{n_i}))}{\lambda_2(\Delta L_{i,n_i+1}(x^{(i)}(k+1)|_{n_i+1}))}$$

$$e_i^- = 1 - \frac{\lambda_2(\Delta L_{i,n_i}(x^{(i)}(k+1)|_{n_i-1}))}{\lambda_2(\Delta L_{i,n_i}(x^{(i)}(k+1)|_{n_i}))}$$

which determine the sub-optimality of the local solutions (34) with  $n_i + 1$  and  $n_i - 1$  with respect to the one obtained with  $n_i$ . In particular,  $e_i^+$  measures the gain, in terms of local algebraic connectivity, one would have by increasing the enlarged neighborhood size from  $n_i$  to  $n_i + 1$ , while  $e_i^-$  measures the loss of local algebraic connectivity going from  $n_i$  to  $n_i - 1$ . (We note that both  $e_i^+$  and  $e_i^-$  are non-negative due to Theorem 7).

Given specific lower/upper thresholds for  $e_i^+$  and  $e_i^-$  the agents can decide locally to increase or decrease  $n_i$  at the successive time step  $k$ , trading off larger communication efforts (for larger  $n_i$ ) to smaller local algebraic connectivity increases (for smaller  $n_i$ ), making Algorithm 1 adaptive. We note that although these sub-optimality measures are local, changing  $n_i$  locally by each agent has an effect on the global solution as illustrated by the relation (38) in Lemma 2. We note also that in order to compute  $e_i^+$  and  $e_i^-$  it is necessary to solve three optimization problems of the kind (25) for each  $i$ . Since this can be computationally expensive, the agents can decide to determine  $e_i^+$  and  $e_i^-$  only once in a given number of discrete time steps.

## 8 Numerical Results

In this section, we present numerical simulation results to illustrate how the proposed distributed algorithm performs with respect to the centralized scheme. We use a benchmark problem motivated by [14]. Our scenario considers  $N = 10$  agents moving on a 2D plane initially placed close to the horizontal axis and forming a connected graph. The initial position vector is  $x_i(0) = [-6.75 + 1.5(i-1), y_i]^T$ , where  $y_i$  is drawn from a Gaussian distribution, with mean 0 and standard deviation  $\sigma = 0.1$ . Randomness is added to test the algorithm's sensitivity to different initial conditions

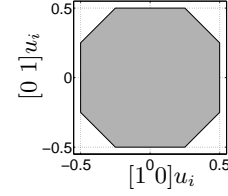


Fig. 3. Polytopic constraint for  $u_i$ . The shaded region represents the set  $\mathcal{U}_i \subset \mathbb{R}^2$ .

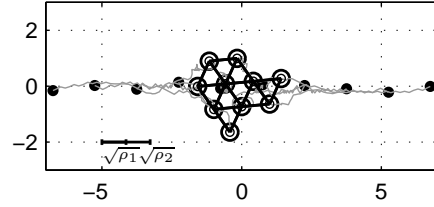


Fig. 4. Centralized solution: the initial positions are marked with black dots. The final positions are marked with circles. The bold lines represent the final communication graph and the thin lines the agent trajectories.

(due to the sequential convex programming approach). We consider the triples  $(A_{1i}, A_{2i}, b_{1i})$  to be all equal to  $(I_2 T_s/2, 0.75 I_2, I_2 T_s/2)$  with  $T_s = 1$  (modeling a discrete-time double integrator dynamics), while all the  $u_i$ 's are constrained in the polytopic region of Figure 3. The other simulation parameters include the weighting function of Figure 1,  $\rho_1 = 0.75$ ,  $\rho_2 = 3$  and final time  $T = 300$ . We performed and analyzed a total of 50 simulation runs.

In Figures 4-5, an example of the trajectories using the centralized and the distributed solutions are depicted. In the adaptive case, we start with  $n_i = 2$  for all agents and at every 5 discrete time step  $k$  we compute the sub-optimality measures. If the gain in increasing the enlarged neighborhood size is high enough, i.e.,  $e_i^+ > 0.05$ , we increase  $n_i$ , while if this gain is not high enough, i.e.,  $e_i^+ < 0.05$ , and the losses in decreasing the neighborhood size are not too big, i.e.,  $e_i^- < 0.01$ , we decrease  $n_i$  to reduce the communication and computation costs.

Figure 6 shows, in the same simulation, the algebraic connectivity as a function of the sampling time  $k$ , and clearly illustrates the nonlinear/non-convex nature of the problem. In fact, in this case, although the distributed approximations are slower to converge than the centralized solution, in the end they achieve a slightly better final  $\lambda_2$ .

Table 1 shows the ratio between the final connectivity of the distributed solution and the centralized one in the 50 simulation runs. For better comparison, we report that in the adaptive case  $n_i = 2.2$  on average, with a maximum of  $n_i = 5$ . We can observe that different choices of the local neighborhood sizes  $n_i$  affect the final achieved  $\lambda_2$ .

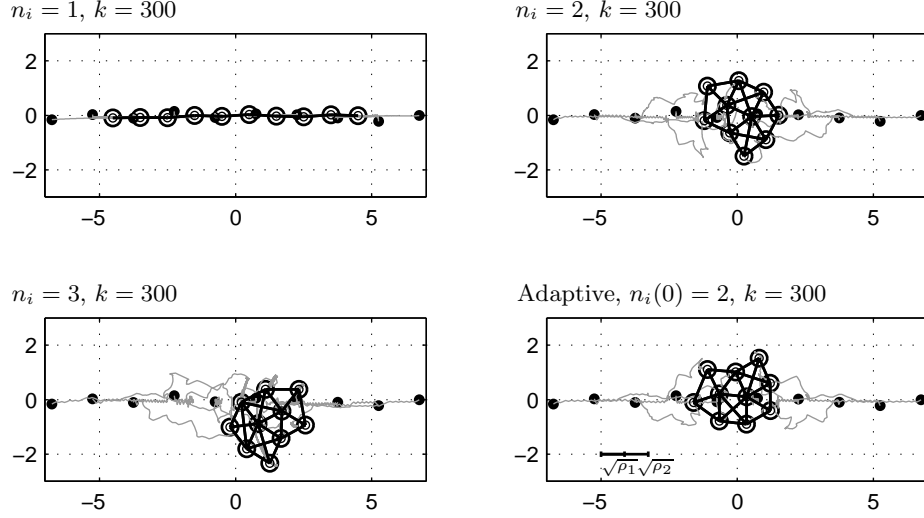


Fig. 5. Simulation results of the distributed approach for various local neighborhood sizes  $n_i$  (same  $\forall i$  except in the adaptive case). The initial positions are marked with black dots. The final positions are marked with circles. The bold lines represent the final communication graph and the thin lines the agent trajectories. In the adaptive case, we start with  $n_i = 2 \forall i$  and at every 5 discrete time step  $k$  we compute  $e_i^+$  and  $e_i^-$ . If  $e_i^+ > 0.05$  we increase  $n_i$ , if  $e_i^- < 0.05$  and  $e_i^- < 0.01$  we decrease  $n_i$ .

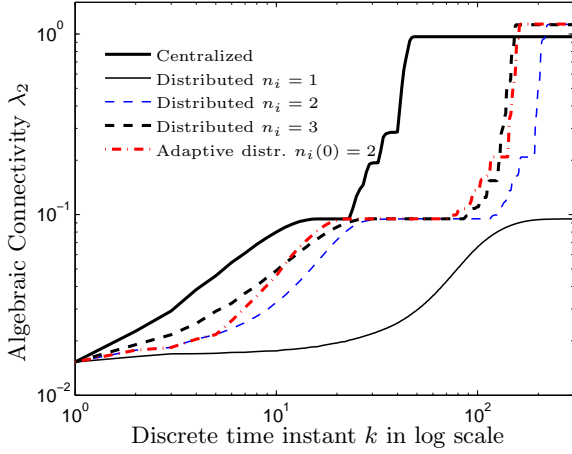


Fig. 6. Algebraic connectivity as a function of time  $k$  for both the centralized and the distributed solutions.

Table 1

Ratio between the final connectivity of the distributed solution and the centralized one. The adaptive case is indicated with  $n_i(0)$ . The cases  $N = \{20, 40\}$  correspond to a random feasible initial configuration (not necessarily a line).

Ratio $\frac{\lambda_2^{\text{distr}}}{\lambda_2^{\text{centr}}}$	$N = 10$				$N = 20$	$N = 40$
	$n_i = 1$	$n_i = 2$	$n_i = 3$	$n_i(0) = 2$	$n_i(0) = 2$	$n_i(0) = 2$
(0.1 – 0.3]	50	26	0	0	0	0
(0.3 – 0.8]	0	0	5	4	3	3
(0.8 – 1.0]	0	12	22	21	21	24
(1.0 – 1.1]	0	12	23	25	26	23

In particular, for the choice  $n_i = 1$ , the agents perform significantly worse than for other  $n_i$ . Furthermore, using the adaptive case, the final  $\lambda_2$  is comparable with the centralized solution in most of the simulations (or even better). This is an important point, since the adaptive case use an enlarged neighborhood size of  $n_i = 2.2$  (on average) and still obtains performances close or better than the fixed choice  $n_i = 3$ .

To further assess the proposed distributed algorithm, we include in Table 1 simulation results for  $N = \{20, 40\}$  robots starting from a feasible random configuration (not necessarily on a line) and using the adaptive algorithm with  $n_i(0) = 2$ . Each of these cases has been run 50 times. We can observe that both in the  $N = 20$  case (where the average  $n_i$  is 2.7) and in the  $N = 40$  case (average  $n_i = 2.6$ ), the results are in line with the conclusions that could be drawn for the case of  $N = 10$ . From these results one could conjecture both the scalability of Algorithm 1 (for the adaptive case) and its increased performances dealing with large systems. In particular, while the number of agents passes from  $N = 10$  to  $N = 40$ , the averaged size of the enlarged neighborhood stays rather the same (and also the performance in term of final  $\lambda_2$ ). This means that the computational and communication efforts for the single agent stay the same. Thus, the gain of the distributed solution with respect of the centralized solution, in terms of computations and communications, increases.

## 9 Conclusions

We have presented a distributed solution to the maximization of the algebraic connectivity of the communication graph in a robotic network. Our characterization can handle more generic LTI agent dynamics than the methods available in the literature and the resulting op-

timization problem is proven to be feasible at each time step under reasonable assumptions. Furthermore the solution can be adjusted based on available resources using local relative sub-optimality measures to aid in adapting the neighborhood size to the agents' needs.

Simulation results confirm the efficacy of our distributed approach and show its practical applicability. Some open issues still remain and will be the focus of our future research. In particular, robustness of the proposed algorithm against estimation errors is currently being investigated. The applicability of the distributed scheme in a broader class of problem formulations involving LMI constraints is part of our research plans, as well as experimental validations.

## References

- [1] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, Belmont, Massachusetts, 1997.
- [2] F. Borrelli, A. Bemporad, and M. Morari. *Predictive Control*. 2011. In preparation, draft available at <http://www.mpc.berkeley.edu/mpc-course-material>.
- [3] S. Boyd. Convex Optimization of Graph Laplacian Eigenvalues. In *Proceedings of the International Congress of Mathematicians*, pages 1311 – 1319, Madrid, Spain, August 2006.
- [4] F. Bullo, J. Cortés, and S. Martínez. *Distributed Control of Robotic Networks*. Applied Mathematics Series. Princeton University Press, 2008.
- [5] D. W. Casbeer, S. Li, R. W. Beard, and R. K. Mehra. Forest Fire Monitoring With Multiple Small UAVs. In *Proceedings of the American Control Conference*, pages 3530 – 3535, Portland, USA, June 2005.
- [6] J. Casper and R. Murphy. Human-Robot Interactions during the Robot-Assisted Urban Search and Rescue Response at the World Trade Center. *IEEE Transaction on Systems, Man, and Cybernetics-Part B: Cybernetics*, 33(6):367 – 385, 2003.
- [7] J. Cortés, S. Martínez, and F. Bullo. Robust Rendezvous for Mobile Autonomous Agents via Proximity Graphs in Arbitrary Dimensions. *IEEE Transaction on Automatic Control*, 51(8):1289 – 1298, 2006.
- [8] N. M. M. de Abreu. Old and new results on algebraic connectivity of graphs. *Linear Algebra and its Applications*, 423(1):53 – 73, 2007.
- [9] M. C. De Gennaro and A. Jadabaie. Decentralized Control of Connectivity for Multi-Agent Systems. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 3628 – 3633, San Diego, USA, December 2006.
- [10] J. Derenick, J. Spletzer, and V. Kumar. A Semidefinite Programming Framework for Controlling Multi-robot Systems in Dynamic Environments. In *Proceedings of the 49th IEEE Conference on Decision and Control*, pages 7172 – 7177, Atlanta, USA, December 2010.
- [11] J. Derenick, J.R. Spletzer, and A. Hsieh. An Optimal Approach to Collaborative Target Tracking with Performance Guarantees. *Journal of Intelligent Robotic Systems*, 56(1):47 – 67, 2009.
- [12] D. Izzo and L. Pettazzi. Autonomous and Distributed Motion Planning for Satellite Swarm. *Journal of Guidance, Control and Dynamics*, 30(2):449 – 459, 2007.
- [13] T. Keviczky and K. H. Johansson. A Study on Distributed Model Predictive Consensus. In *Proceedings of 17th IFAC World Congress*, pages 1516 – 1521, Seoul, Korea, July 2008.
- [14] Y. Kim and M. Mesbahi. On Maximizing the Second Smallest Eigenvalue of a State-Dependent Graph Laplacian. *IEEE Transactions of Automatic Control*, 51(1):116 – 120, 2006.
- [15] C. Langbort, L. Xiao, R. D'Andrea, and S. Boyd. A Decomposition Approach to Distributed Analysis of Networked Systems. In *Proceedings of the 43rd IEEE Conference on Decision and Control*, pages 3980 – 3985, Paradise Island, Bahamas, December 2004.
- [16] H. Y. K. Lau and A. W. Y. Ko. Coordination of Cooperative Search and Rescue Robots for Disaster Relief. In *Proceedings of 17th IFAC World Congress*, pages 895 – 900, Seoul, Korea, July 2008.
- [17] N. E. Leonard, D. A. Paley, R. E. Davis, D. M. Fratantoni, F. Lekien, and F. Zhang. Coordinated Control of an Underwater Glider Fleet in an Adaptive Ocean Sampling Field Experiment in Monterey Bay. *Journal of Field Robotics*, 27(6):718 – 740, 2010.
- [18] G. Mathews and H. Durrant-Whyte. Decentralised Optimal Control for Reconnaissance. In *Proceedings of the Conference on Information, Decision and Control*, pages 314 – 319, Adelaide, Australia, February 2007.
- [19] R. Olfati-Saber and R.M. Murray. Consensus Problems in Networks of Agents with Switching Topology and Time-Delays. *IEEE Transactions on Automatic Control*, 49(9):1520 – 1533, 2004.
- [20] A. Rantzer. Dynamic Dual Decomposition for Distributed Control. In *Proceedings of the American Control Conference*, pages 884 – 888, St. Louis, USA, June 2009.
- [21] W. Ren and R.W. Beard. *Distributed Consensus in Multi-Vehicle Cooperative Control: Theory and Applications*. Springer-Verlag London, 2008.
- [22] M. Schuresko and J. Cortés. Distributed Motion Constraints for Algebraic Connectivity of Robotic Networks. *Journal of Intelligent Robotics Systems*, 56(1):99 – 126, 2009.
- [23] A. Simonetto and T. Keviczky. Distributed Multi-Target Tracking via Mobile Robotic Networks: a Localized Non-iterative SDP Approach. In *Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference*, pages 4226 – 4231, Orlando, USA, December 2011.
- [24] A. Simonetto, T. Keviczky, and R. Babuška. On Distributed Algebraic Connectivity Maximization in Robotic Networks. In *Proceedings of the American Control Conference*, pages 2180 – 2185, San Francisco, USA, June – July 2011.
- [25] D. Spanos and R. M. Murray. Robust Connectivity of Networked Vehicles. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 2893 – 2898, Paradise Island, Bahamas, December 2004.
- [26] P. Yang, R. A. Freeman, G. J. Gordon, K. M. Lynch, S. S. Srinivasa, and R. Sukthankar. Decentralized Estimation and Control of Graph Connectivity in Mobile Sensor Networks. In *Proceedings of the American Control Conference*, pages 2678 – 2683, Seattle, USA, June 2008.
- [27] M. M. Zavlanos and G. J. Pappas. Distributed Connectivity Control of Mobile Networks. *IEEE Transactions on Robotics*, 24(6):1416 – 1428, 2008.

- [28] M. M. Zavlanos, A. Ribeiro, and G. J. Pappas. Distributed Control of Mobility and Routing in Networks of Robots. In *Proceedings of the 12th IEEE International Workshop on Signal Processing Advances in Wireless Communications*, pages 236 – 240., San Francisco, USA, June 2011.
- [29] M. M. Zavlanos, H. G. Tanner, and A. Jadbabaie. Hybrid Control for Connectivity Preserving Flocking. *IEEE Transactions on Automatic Control*, 54(12):2869 – 2875, 2009.

## A Generalization to LTI systems

This appendix considers the issues related to the use of more general systems than (14) in the centralized problem (22). We start from a generalization of (14) considering the  $(M + 2)$ -order system:

$$\begin{pmatrix} x_i(\tau + 1) \\ v_i(\tau + 1) \\ y_{1i}(\tau + 1) \\ \vdots \\ y_{Mi}(\tau + 1) \end{pmatrix} = \begin{pmatrix} I_3 & \star & \star & \cdots \\ 0_3 & \star & \star & \cdots \\ 0_3 & \star & \star & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} \begin{pmatrix} x_i(\tau) \\ v_i(\tau) \\ y_{1i}(\tau) \\ \vdots \\ y_{Mi}(\tau) \end{pmatrix} + \begin{pmatrix} 0_3 \\ \vdots \\ b_{1i}I_3 \end{pmatrix} u_i(\tau)$$

where the stars represent non-zero elements and  $b_{1i} \in \mathbb{R}_0$ . It is not difficult to see that Algorithm 1 is also applicable to these types of systems, under quite general assumptions and minor modifications. The key idea is to compute the control actions every  $M + 2$  steps while the crucial drawback is that the larger  $M + 2$  is, the more  $\rho_1$  has to be shrunk to accommodate the collision avoidance requirement (see condition (21) which has to be generalized in this case in a straightforward manner).

Consider now the generic LTI system

$$\mathbf{x}_i(\tau + 1) = A_i \mathbf{x}_i(\tau) + B_i u_i(\tau)$$

where the couple  $(A_i, B_i)$  is controllable and where the state can be partitioned as  $(x_i(\tau)^\top, \xi_i(\tau)^\top)^\top$ . In order to apply Algorithm 1, we need to characterize a modification of the set  $\mathcal{F}_i$  which is defined as:

$$\mathbf{x}_i(\tau) \in \mathcal{F}_i^T \Rightarrow \exists \{u_i(\tau), \dots, u_i(\tau + T - 1)\} \in \bar{\mathcal{U}}_i \text{ such that } A_i^T \mathbf{x}_i(\tau) + \sum_{h=0}^{T-1} A_i^{T-1-h} B_i u_i(\tau + h) = (x_i(\tau)^\top, 0)^\top, \forall \tau \in \mathbb{N}_+$$

By computing  $\mathcal{F}_i^T$ , we can extend Algorithm 1 also to general LTI systems, calculating the control every  $T$  time steps. However, several issues have to be addressed: (i) the parameter  $T$  is agent-dependent and it depends on  $\tau$ , making the determination of a single  $\mathcal{F}_i^T$  quite complex; (ii) the set  $\mathcal{F}_i^T$  depends also on the position  $x_i(\tau)$  restricting the area in which the agents can move; (iii) since  $T$  can be in general quite large, the condition on  $\rho_1$  could be rather limiting and it could conflict with the requirements on  $\mathcal{F}_i^T$ .

## B Proof of Theorem 2

At optimality the local constraints for the subproblem  $\Delta \mathbf{P}_p$  in (25) are the following:

$$\begin{aligned} \forall p | p \in \mathcal{J}_i^* \cap \mathcal{J}_j^* : \\ \hat{Q}_{2.1} : \quad \Delta f_d(x_i(k+1), x_j(k+1)) = \\ d_{ij}^2(k) + [ij]^\top (\delta \hat{x}_{pi}(k+1) - \delta \hat{x}_{pj}(k+1)) > \hat{\rho}_{1ij}, \end{aligned} \quad (\text{B.1a})$$

$$\forall p | p \in \mathcal{J}_i^* : \quad \hat{Q}_{2.2} : \quad \hat{\mathbf{x}}_{pi}(k+1) \in \hat{\mathcal{F}}_i = \mathcal{F}_i \quad (\text{B.1b})$$

$$\hat{Q}_{2.3} : \quad \hat{\mathbf{u}}_{pi}(k) \in \hat{\mathcal{U}}_i \quad (\text{B.1c})$$

$$\hat{Q}_{2.4} : \quad \hat{\mathbf{x}}_{pi}(k+1) = \hat{\mathcal{D}}_i(\mathbf{x}_i(k), \hat{\mathbf{u}}_{pi}(k)) \quad (\text{B.1d})$$

The theorem claims that using the specified choice for  $\mathcal{S}_{\Delta \hat{Q}_{2i}}$ , if we combine the local optimal solutions  $(\hat{\mathbf{x}}_{pi}(k+1), \hat{\mathbf{u}}_{pi}(k))$  which satisfy the local constraints (B.1), using the positive linear combinations (26), (31), and (32) we will obtain a couple  $(\mathbf{x}(k+1), \mathbf{u}(k))$  that satisfies the constraint  $\Delta Q_2$  of the global problem (22). This is what we need to prove.

Consider  $\hat{Q}_{2.1}$  in (B.1a) and the positive linear combination for  $x(k+1)$  in (26). By Lemma 1 follows:

$$\begin{aligned} d_{ij}^2(k) + [ij]^\top (\delta x_i(k+1) - \delta x_j(k+1)) = \\ = d_{ij}^2(k) + \sum_{p \in \mathcal{J}_i^* \cap \mathcal{J}_j^*} [ij]^\top \alpha_p (\delta \hat{x}_{pi}(k+1) - \delta \hat{x}_{pj}(k+1)) > \\ (1 - \bar{\alpha}_{ij}) d_{ij}^2(k) + \bar{\alpha}_{ij} \hat{\rho}_{1ij} \end{aligned} \quad (\text{B.2})$$

For  $x(k+1)$  it is required the satisfaction of the global constraint:

$$d_{ij}^2(k) + [ij]^\top (\delta x_i(k+1) - \delta x_j(k+1)) > \rho_1 \quad (\text{B.3})$$

which can be accomplished by selecting  $\hat{\rho}_{1ij}$  such that:

$$(1 - \bar{\alpha}_{ij}) d_{ij}^2(k) + \bar{\alpha}_{ij} \hat{\rho}_{1ij} = \rho_1 \quad (\text{B.4})$$

This gives the formula for  $\hat{\rho}_{1ij}$  in (30).

Consider the constraints  $\hat{Q}_{2.4}$  in (B.1d) on the agents' dynamics. For the positive linear combination (26) the combined system dynamics becomes

$$\begin{aligned} \left( \sum_{p \in \mathcal{J}_i^*} \alpha_p \hat{v}_{pi}(k+1) \right) = \begin{pmatrix} I_3 & \bar{\alpha}_i \hat{A}_{1i} (I_3 + \hat{A}_{2i}) \\ 0_3 & \bar{\alpha}_i \hat{A}_{2i}^2 \end{pmatrix} \begin{pmatrix} x_i(k) \\ v_i(k) \end{pmatrix} + \\ \begin{pmatrix} \hat{b}_{1i} \hat{A}_{1i} & 0_3 \\ \hat{b}_{1i} \hat{A}_{2i} & \hat{b}_{1i} I_3 \end{pmatrix} \sum_{p \in \mathcal{J}_p^*} \alpha_p \hat{\mathbf{u}}_{pi}(k) \end{aligned} \quad (\text{B.5})$$

Since the agents have to move according to the dynamical system (16) encoded in the global constraint  $Q_{2.4}$

of (22), the update (B.5) and the state equation (16) have to be the same. It is not difficult to see that this is ensured by the choice  $\hat{A}_{1i} = \bar{\alpha}_i^{-1} A_{1i}$ ,  $\hat{A}_{2i} = A_{2i}$ ,  $\hat{b}_{1i} = \bar{\alpha}_i b_{1i}$ , and the linear combinations (31) and (32) for the local control inputs  $\hat{\mathbf{u}}_{pi}(k)$  and local velocities  $\hat{v}_{pi}(k+1)$ .

From the linear combination on the control (31) and the global constraint  $\mathcal{Q}_{2.3}$  in (22) follows the specification for the local constraint  $\hat{\mathcal{Q}}_{2.3}$  in (25):

$$\hat{\mathcal{U}}_i = \{\hat{\mathbf{u}}_{pi}(k) \in \mathbb{R}^3 | H_i \hat{\mathbf{u}}_{pi} \leq \bar{\alpha}_i^{-1} h_i\} \quad (\text{B.6})$$

from which  $(\hat{H}_i, \hat{h}_i) = (H_i, \bar{\alpha}_i^{-1} h_i)$ . We recall that the positive linear combination on the control input (31) has been constructed in a way to steer the system (16) from the position  $x(k)$  to the updated position  $x(k+1)$  in (26) while respecting the global constraints  $\mathcal{Q}_{2.3}$  in (22).

Consider now  $\hat{\mathcal{Q}}_{2.2}$  in (25). We need to prove that if the local optimal states  $\hat{\mathbf{x}}_{pi}(k+1)$  belong to the set  $\hat{\mathcal{F}}_i$  in (25), then the updated state  $\mathbf{x}_i(k+1)$  constructed via the linear combinations on position (26) and velocity (32) belongs to the set  $\mathcal{F}_i$  as expressed in the global constraint  $\mathcal{Q}_{2.2}$  in (22). First of all, it is straightforward to see that the local inequalities

$$- \begin{pmatrix} \hat{H}_i \hat{b}_{1i}^{-1} (I_3 + \hat{A}_{2i}) \\ \hat{H}_i \hat{b}_{1i}^{-1} \hat{A}_{2i} (I_3 + 2\hat{A}_{2i}) \end{pmatrix} \hat{v}_{pi}(k+1) \leq \begin{pmatrix} \hat{h}_i \\ \hat{h}_i \end{pmatrix} \quad (\text{B.7})$$

are equivalent to the inequalities (20), meaning that by construction  $\hat{\mathcal{F}}_i = \mathcal{F}_i$ . Recall that the set  $\mathcal{F}_i$  does not constrain the position. Since the updated velocity  $v_i(k+1)$  in (32) is obtained by a positive linear combination of local  $\hat{v}_{pi}(k+1)$  then also  $v_i(k+1)$  will satisfy the inequalities (B.7), and therefore the updated state  $\mathbf{x}_i(k+1)$  belongs to  $\mathcal{F}_i$ .

Having ensured that with the choices of Theorem 2 the positive linear combinations of the local solutions satisfy the constraints  $\mathcal{Q}_{2.1} - \mathcal{Q}_{2.4}$  of (22), Theorem 2 is proven.  $\square$